

Práctica 1.4. Protocolo IPv6

Objetivos

En esta práctica se estudian los aspectos básicos del protocolo IPv6, el manejo de los diferentes tipos de direcciones y mecanismos de configuración. Además se analizarán las características más importantes del protocolo ICMP versión 6.



Activar el **portapapeles bidireccional** (menú Dispositivos) en las máquinas virtuales.

Usar la opción de Virtualbox (menú Ver) para realizar **capturas de pantalla**.

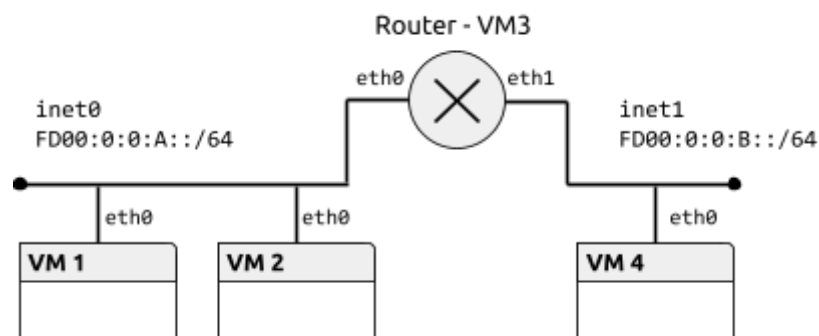
La **contraseña** del usuario cursoredes es cursoredes.

Contenidos

- Preparación del entorno para la práctica
- Direcciones de enlace local
- Direcciones ULA
- Encaminamiento estático
- Configuración persistente
- Autoconfiguración. Anuncio de prefijos
- ICMPv6

Preparación del entorno para la práctica

Configuraremos la topología de red que se muestra en la siguiente figura:



El fichero de configuración de la topología tendría el siguiente contenido:

```
netprefix inet
machine 1 0 0
machine 2 0 0
machine 3 0 0 1 1
machine 4 0 1
```

Direcciones de enlace local

Una dirección de enlace local es únicamente válida en la subred que está definida. Ningún encaminador dará salida a un datagrama con una dirección de enlace local como destino. El prefijo de formato para estas direcciones es fe80::/10.

Ejercicio 1 [VM1, VM2]. Activar el interfaz eth0 en VM1 y VM2. Comprobar las direcciones de enlace local que tienen asignadas con el comando ip.

VM1: fe80::a00:27ff:feb8:ad92/64

VM2 : fe80::a00:27ff:feea:9bf2/64

Ejercicio 2 [VM1, VM2]. Comprobar la conectividad entre VM1 y VM2 con la orden ping6 (o ping -6). Cuando se usan direcciones de enlace local, y **sólo en ese caso**, es necesario especificar el interfaz origen, añadiendo %<nombre_interfaz> a la dirección. Consultar las opciones del comando ping6 en la página de manual. Observar el tráfico generado con Wireshark, especialmente los protocolos encapsulados en cada datagrama y los parámetros del protocolo IPv6.

Copiar el comando utilizados y su salida. Copiar una captura de pantalla de Wireshark donde se vean los campos de la cabecera IPv6.

En VM1: ping6 -c 1 fe80::a00:27ff:feea:9bf2 -I eth0 (%eth0 no me funcionaba bien)

PING fe80::a00:27ff:feea:9bf2 (fe80::a00:27ff:feea:9bf2) from fe80::a00:27ff:feb8:ad92%eth0 eth0: 56 data bytes

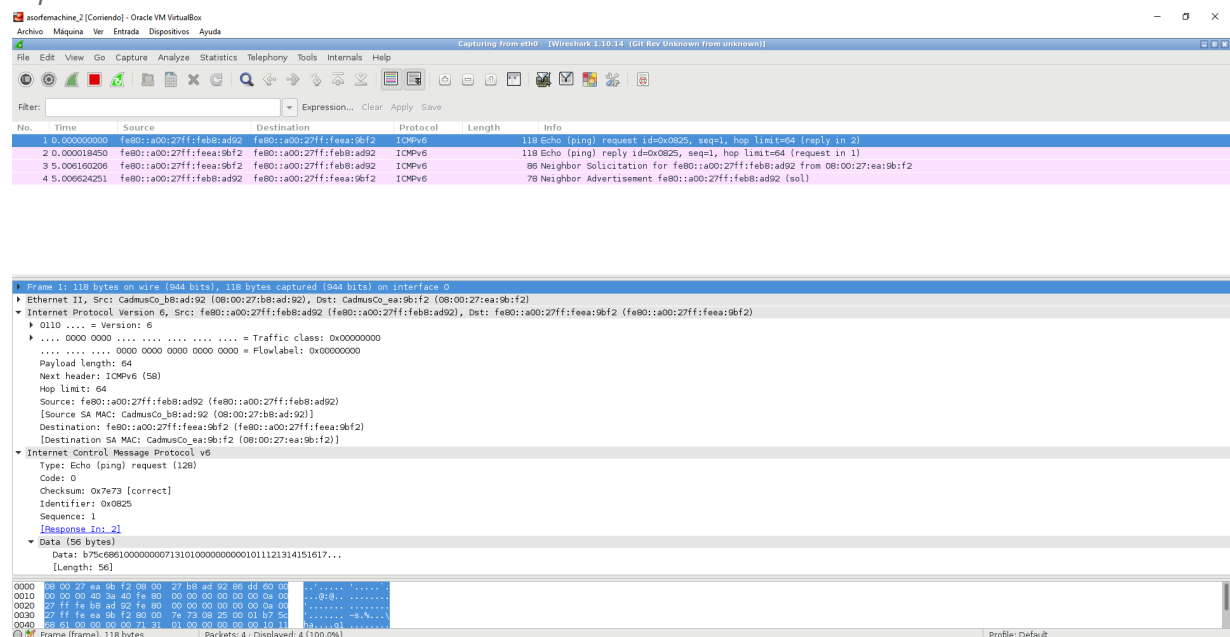
64 bytes from fe80::a00:27ff:feea:9bf2%eth0: icmp_seq=1 ttl=64 time=0.465 ms

--- fe80::a00:27ff:feea:9bf2 ping statistics ---

1 packets transmitted, 1 received, 0% packet loss, time 0ms

rtt min/avg/max/mdev = 0.465/0.465/0.465/0.000 ms

Captura de VM2



Neighbor solicitation

Echo request

Echo reply -Ver 0110(6)

-traffic class y flow label 0

-payload length 64

-next header icmpv6

-hop limit 64

-source and destination address

Ejercicio 3 [Router, VM4]. Activar el interfaz de VM4 y los dos interfaces de Router. Comprobar la conectividad entre Router y VM1, y entre Router y VM4 usando la dirección de enlace local.

Copiar los comandos utilizados y su salida.

En **VM3**:

```
ip link set eth0 up : fe80::a00:27ff:fe9b:e8c0/64
```

```
ip link set eth1 up: fe80::a00:27ff:fe49:1476/64
```

En **VM4**:

```
ip link set eth0 up: fe80::a00:27ff:fe6b:8f45/64
```

En **VM1**:

```
ping6 -c 1 fe80::a00:27ff:fe9b:e8c0 -I eth0
```

```
PING fe80::a00:27ff:fe9b:e8c0(fe80::a00:27ff:fe9b:e8c0) from fe80::a00:27ff:feb8:ad92%eth0 eth0: 56 data bytes
```

```
64 bytes from fe80::a00:27ff:fe9b:e8c0%eth0: icmp_seq=1 ttl=64 time=0.610 ms
```

```
--- fe80::a00:27ff:fe9b:e8c0 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 0.610/0.610/0.610/0.000 ms
```

En **VM3**:

```
ping6 -c 1 fe80::a00:27ff:fe6b:8f45 -I eth1
```

```
PING fe80::a00:27ff:fe6b:8f45(fe80::a00:27ff:fe6b:8f45) from fe80::a00:27ff:fe49:1476%eth1 eth1: 56 data bytes
```

```
64 bytes from fe80::a00:27ff:fe6b:8f45%eth1: icmp_seq=1 ttl=64 time=0.790 ms
```

```
--- fe80::a00:27ff:fe6b:8f45 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 0.790/0.790/0.790/0.000 ms
```

Para saber más... En el protocolo IPv4 también se reserva el bloque 169.254.0.0/16 para direcciones de enlace local, cuando no es posible la configuración de los interfaces por otras vías. Los detalles se describen en el RFC 3927.

Direcciones ULA

Una dirección ULA (*Unique Local Address*) puede usarse dentro de una organización, de forma que los encaminadores internos del sitio deben encaminar los datagramas con una dirección ULA como destino. El prefijo de formato para estas direcciones es fc00::/7.

Ejercicio 4 [VM1, VM2]. Configurar VM1 y VM2 para que tengan una dirección ULA en la red fd00:0:0:a::/64 con el comando ip. La parte de identificador de interfaz puede elegirse libremente, siempre que no coincida para ambas máquinas. Incluir la longitud del prefijo al fijar las direcciones.

Copiar los comandos utilizados.

En **VM1**:

```
ip address add fd00:0:0:a::1/64 dev eth0
```

En **VM2**:

```
ip address add fd00:0:0:a::2/64 dev eth0
```

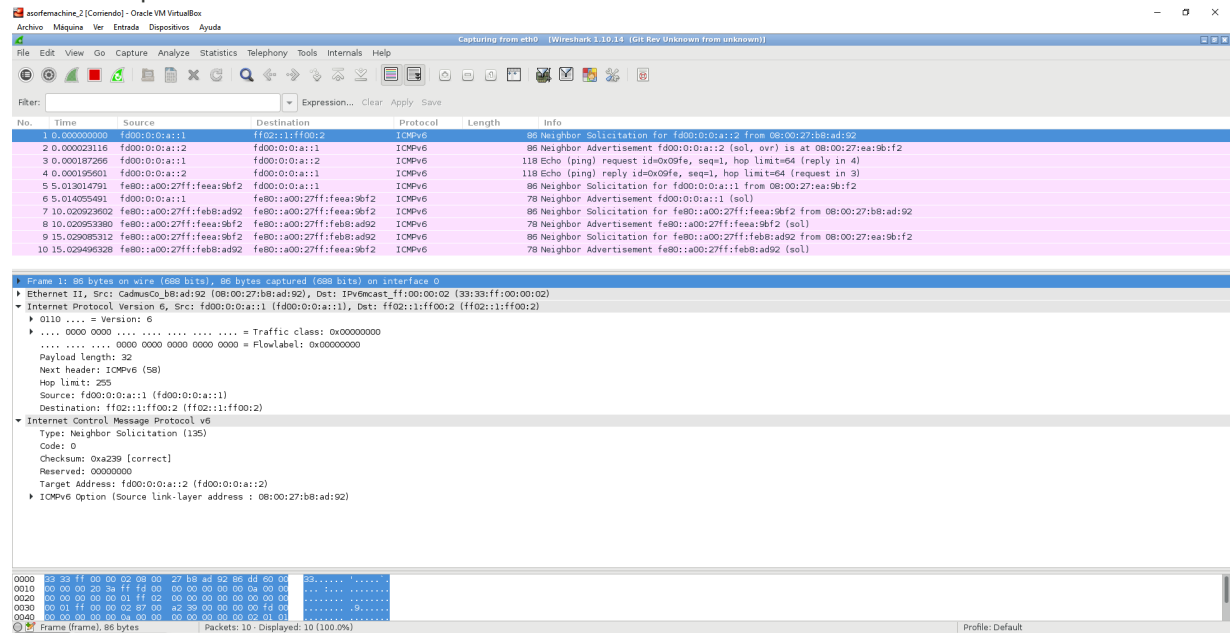
Ejercicio 5 [VM1, VM2]. Comprobar la conectividad entre VM1 y VM2 con la orden ping6 usando la

nueva dirección. Observar los mensajes intercambiados con Wireshark.

```
En VM1: ping6 -c 1 fd00:0:0:a::2 -I eth0
PING fd00:0:0:a::2(fd00:0:0:a::2) from fd00:0:0:a::1 eth0: 56 data bytes
64 bytes from fd00:0:0:a::2: icmp_seq=1 ttl=64 time=0.437 ms
```

```
--- fd00:0:0:a::2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.437/0.437/0.437/0.000 ms
```

En VM2 captura wireshark



Ejercicio 6 [Router, VM4]. Configurar direcciones ULA en los dos interfaces de Router (redes fd00:0:0:a::/64 y fd00:0:0:b::/64) y en el de VM4 (red fd00:0:0:b::/64). Elegir el identificador de interfaz de forma que no coincida dentro de la misma red.

Copiar los comandos utilizados.

En VM3:

```
ip address add fd00:0:0:a::3/64 dev eth0
```

```
ip address add fd00:0:0:b::1/64 dev eth1
```

En VM4:

```
ip address a fd00:0:0:b::2/64 dev eth0
```

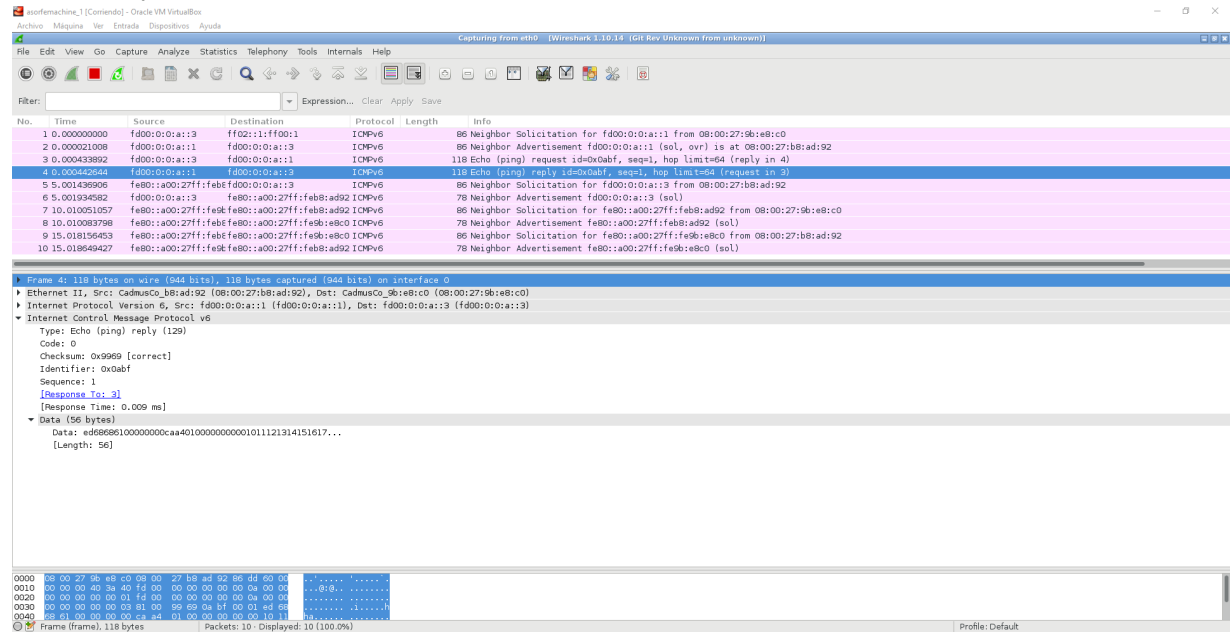
Ejercicio 7 [Router]. Comprobar la conectividad entre Router y VM1, y entre Router y VM4 usando direcciones ULA. Comprobar además que VM1 no puede alcanzar a VM4.

Copiar los comandos utilizados.

```
En VM3: ping6 -c 1 fd00:0:0:a::1
PING fd00:0:0:a::1(fd00:0:0:a::1) 56 data bytes
64 bytes from fd00:0:0:a::1: icmp_seq=1 ttl=64 time=0.856 ms
```

```
--- fd00:0:0:a::1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.856/0.856/0.856/0.000 ms
```

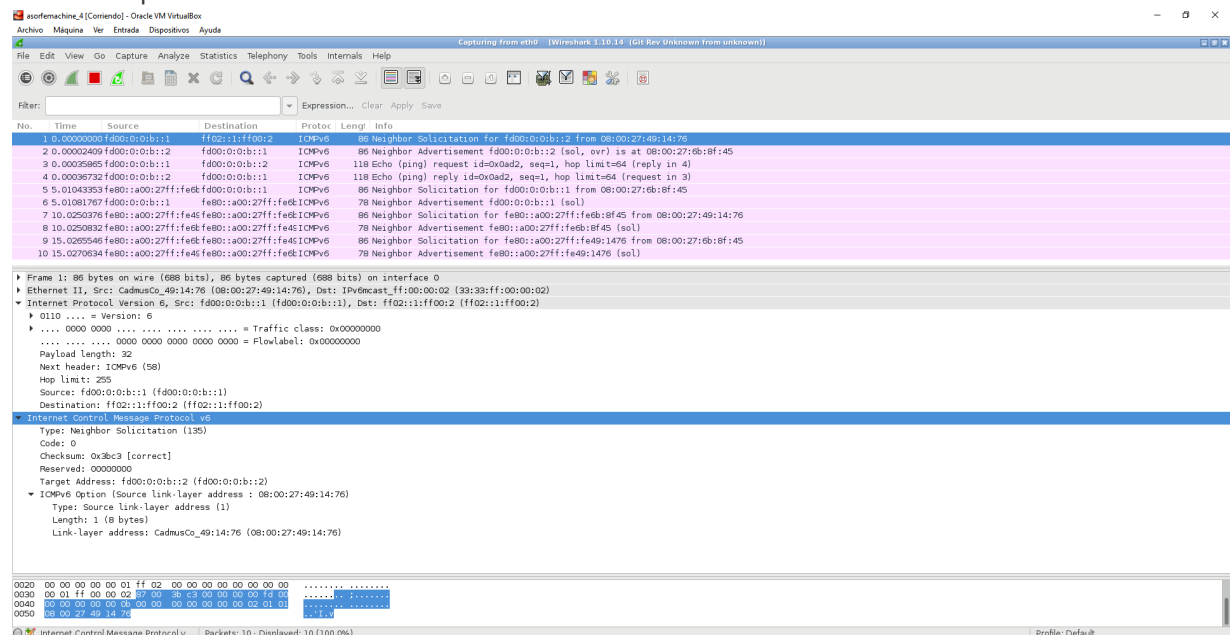
En VM1 captura wireshark



```
En VM3: ping6 -c 1 fd00:0:0:b::2
PING fd00:0:0:b::2(fd00:0:0:b::2) 56 data bytes
64 bytes from fd00:0:0:b::2: icmp_seq=1 ttl=64 time=0.776 ms
```

```
--- fd00:0:0:b::2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.776/0.776/0.776/0.000 ms
```

En VM4 captura wireshark



En VM1: **ping6 -c 1 fd00:0:0:b::2**
connect: Network is unreachable

Encaminamiento estático

Según la topología que hemos configurado en esta práctica, Router debe encaminar el tráfico entre las redes fd00:0:0:a::/64 y fd00:0:0:b::/64. En esta sección vamos a configurar un encaminamiento estático basado en las rutas que fijaremos manualmente en todas las máquinas.

Ejercicio 8 [VM1, Router]. Consultar las tablas de rutas en VM1 y Router con el comando `ip route`. Consultar la página de manual del comando para seleccionar las rutas IPv6.

Copiar los comandos utilizados y su salida.

En VM1 : **route -6**

Kernel IPv6 routing table

Destination	Next Hop	Flag	Met	Ref	Use	If
::/96	::	!n	1024	0	0	lo
0.0.0.0/96	::	!n	1024	0	0	lo
2002:a00::/24	::	!n	1024	0	0	lo
2002:7f00::/24	::	!n	1024	0	0	lo
2002:a9fe::/32	::	!n	1024	0	0	lo
2002:ac10::/28	::	!n	1024	0	0	lo
2002:c0a8::/32	::	!n	1024	0	0	lo
2002:e000::/19	::	!n	1024	0	0	lo
3ffe:ffff::/32	::	!n	1024	0	0	lo
fd00:0:0:a::/64	::	U	256	1	3	eth0
fe80::/64	::	U	256	1	15	eth0
::/0	::	!n	-1	1	21	lo
localhost/128	::	Un	0	2	27	lo
localhost.localdomain/128	::	Un	0	2	4	lo
localhost.localdomain/128	::	Un	0	2	22	lo
ff00::/8	::	U	256	1	1	eth0
::/0	::	!n	-1	1	21	lo

En VM3: **route -6**

Kernel IPv6 routing table

Destination	Next Hop	Flag	Met	Ref	Use	If
::/96	::	!n	1024	0	0	lo
0.0.0.0/96	::	!n	1024	0	0	lo
2002:a00::/24	::	!n	1024	0	0	lo
2002:7f00::/24	::	!n	1024	0	0	lo
2002:a9fe::/32	::	!n	1024	0	0	lo
2002:ac10::/28	::	!n	1024	0	0	lo
2002:c0a8::/32	::	!n	1024	0	0	lo
2002:e000::/19	::	!n	1024	0	0	lo
3ffe:ffff::/32	::	!n	1024	0	0	lo
fd00:0:0:a::/64	::	U	256	1	2	eth0
fd00:0:0:b::/64	::	U	256	1	2	eth1
fe80::/64	::	U	256	0	0	eth0
fe80::/64	::	U	256	0	0	eth1
::/0	::	!n	-1	1	135	lo
localhost/128	::	Un	0	2	27	lo
localhost.localdomain/128	::	Un	0	2	3	lo
localhost.localdomain/128	::	Un	0	2	3	lo

```
localhost.localdomain/128 [::] Un 0 2 2 lo
localhost.localdomain/128 [::] Un 0 2 2 lo
ff00::/8 [::] U 256 1 1 eth0
ff00::/8 [::] U 256 0 0 eth1
[::]/0 [::] !n -1 1 135 lo
```

Ejercicio 9 [Router]. Para que Router actúe efectivamente como encaminador, hay que activar el reenvío de paquetes (*packet forwarding*). De forma temporal, se puede activar con el comando `sysctl -w net.ipv6.conf.all.forwarding=1`.

Ejercicio 10 [VM1, VM2, VM4]. Finalmente, hay que configurar la tabla de rutas en las máquinas virtuales. Añadir la dirección correspondiente de Router como ruta por defecto con el comando `ip route`. Comprobar la conectividad entre VM1 y VM4 usando el comando `ping6`.

Copiar los comandos utilizados y su salida.

En VM1y VM2:

```
ip -6 route add fd00:0:0:b::/64 via fd00:0:0:a::3
```

En VM4:

```
ip -6 route add fd00:0:0:a::/64 via fd00:0:0:b::1
```

En VM1:

```
ping6 -c 1 fd00:0:0:b::2
```

```
PING fd00:0:0:b::2(fd00:0:0:b::2) 56 data bytes
```

```
64 bytes from fd00:0:0:b::2: icmp_seq=1 ttl=63 time=0.789 ms
```

```
--- fd00:0:0:b::2 ping statistics ---
```

```
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 0.789/0.789/0.789/0.000 ms
```

Ejercicio 11 [VM1, Router, VM4]. Abrir Wireshark en Router e iniciar dos capturas, una en cada interfaz de red. Borrar la tabla de vecinos en VM1 y Router (con `ip neigh flush dev <interfaz>`). Usar la orden `ping6` entre VM1 y VM4. Completar la siguiente tabla con todos los mensajes hasta el primer ICMP Echo Reply:

Red fd00:0:0:a::/64 - Router (eth0)

MAC Origen	MAC Destino	IPv6 Origen	IPv6 Destino	ICMPv6 Tipo
08:00:27:b8:ad:92	33:33:ff:00:00:03	fd00:0:0:a::1	ff02:1:ff00:3	Neighbor solicitation
08:00:27:9b:e8:c0	08:00:27:b8:ad:92	fd00:0:0:a::3	fd00:0:0:a::1	Neighbor advertisement
08:00:27:b8:ad:92	08:00:27:9b:e8:c0	fd00:0:0:a::1	fd00:0:0:b::2	Echo request
08:00:27:9b:e8:c0	08:00:27:b8:ad:92	fd00:0:0:b::2	fd00:0:0:a::1	Echo reply

Red fd00:0:0:b::/64 - Router (eth1)

MAC Origen	MAC Destino	IPv6 Origen	IPv6 Destino	ICMPv6 Tipo
08:00:27:49:14:76	33:33:ff:00:00:02	fe80::a00:27ff:fe49:1476	ff02:1:ff00:2	Neighbor solicitation

08:00:27:6b:8f:45	08:00:27:49:14:76	fd00:0:0:b::2	fe80::a00:27ff:fe49:1476	Neighbor advertisement
08:00:27:49:14:76	08:00:27:6b:8f:45	fd00:0:0:a::1	fd00:0:0:b::2	Echo request
08:00:27:6b:8f:45	08:00:27:49:14:76	fd00:0:0:b::2	fd00:0:0:a::1	Echo reply

Copiar dos capturas de pantalla de Wireshark.

Captura *eth0*

The screenshot shows a Wireshark capture of the *eth0* interface. The packet list displays several ICMPv6 messages:

- Neighbor Solicitation for fd00:0:0:a::3 from 08:00:27:b8:ad:92
- Neighbor Advertisement fd00:0:0:a::3 (rtr, sol, ovr) is at 08:00:27:9b:e8:c0
- Neighbor Solicitation for fd00:0:0:a::3 from 08:00:27:b8:ad:92
- Neighbor Advertisement fd00:0:0:a::3 (rtr, sol, ovr) is at 08:00:27:9b:e8:c0
- Neighbor Solicitation for fd00:0:0:a::3 from 08:00:27:b8:ad:92
- Neighbor Advertisement fd00:0:0:a::3 (rtr, sol, ovr) is at 08:00:27:9b:e8:c0
- Echo (ping) request id=0x0914, seq=1, hop limit=64 (reply in 8)
- Echo (ping) reply id=0x0914, seq=1, hop limit=63 (request in 7)
- Neighbor Solicitation for fd00:0:0:a::1 from 08:00:27:9b:e8:c0
- Neighbor Advertisement fd00:0:0:a::1 (sol)
- Neighbor Solicitation for fe80::a00:27ff:feb8:ad:92 from 08:00:27:9b:e8:c0
- Neighbor Advertisement fd00:0:0:a::1 (sol)
- Neighbor Solicitation for fe80::a00:27ff:feb8:ad:92 (sol)
- Neighbor Solicitation for fe80::a00:27ff:fe9b:e8:c0 from 08:00:27:b8:ad:92
- Neighbor Advertisement fe80::a00:27ff:fe9b:e8:c0 (rtr, sol)

The packet details pane shows the selected packet (Frame 8) as an Internet Control Message Protocol v6 message.

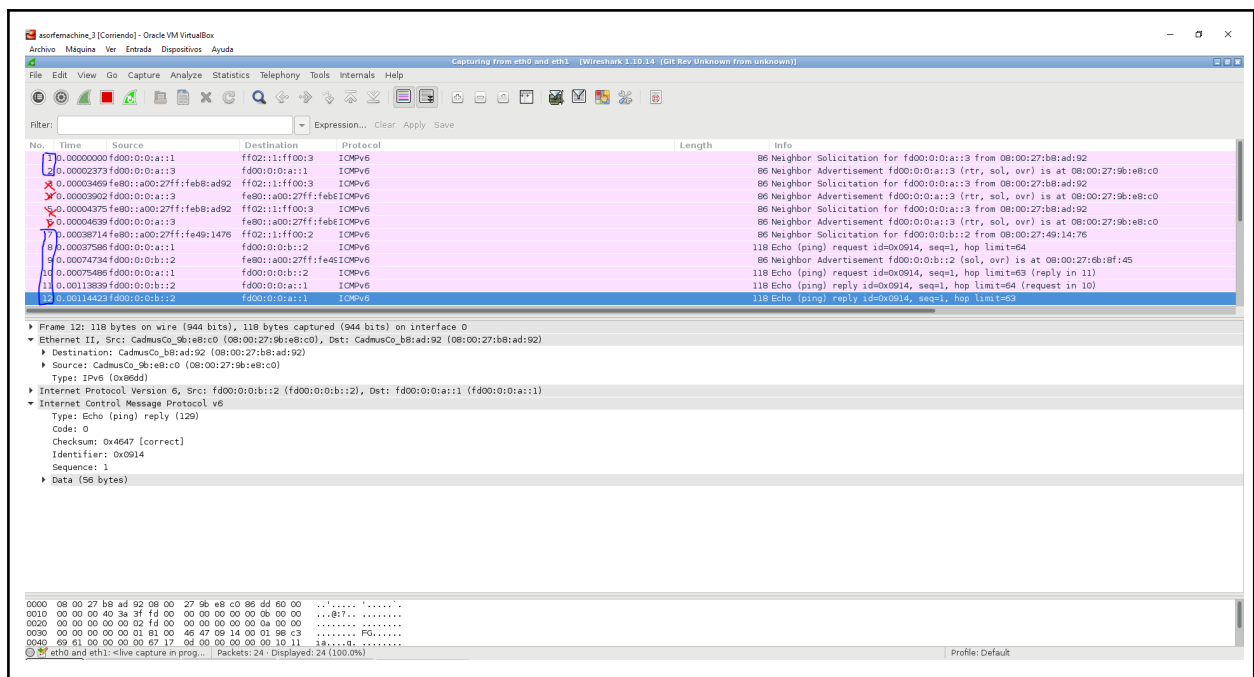
Captura *eth1*

The screenshot shows a Wireshark capture of the *eth1* interface. The packet list displays several ICMPv6 messages:

- Neighbor Solicitation for fd00:0:0:b::2 from 08:00:27:49:14:76
- Neighbor Advertisement fd00:0:0:b::2 (sol, ovr) is at 08:00:27:6b:8f:45
- Echo (ping) request id=0x0914, seq=1, hop limit=63 (reply in 4)
- Echo (ping) reply id=0x0914, seq=1, hop limit=64 (request in 3)
- Neighbor Solicitation for fe80::a00:27ff:fe49:1476 from 08:00:27:6b:8f:45
- Neighbor Advertisement fe80::a00:27ff:fe49:1476 (rtr, sol)
- Neighbor Solicitation for fd00:0:0:b::1 from 08:00:27:6b:8f:45
- Neighbor Advertisement fd00:0:0:b::1 (rtr, sol)
- Neighbor Solicitation for fe80::a00:27ff:fe6b:8f:45 from 08:00:27:49:14:76
- Neighbor Advertisement fe80::a00:27ff:fe6b:8f:45 (sol)

The packet details pane shows the selected packet (Frame 4) as an Internet Control Message Protocol v6 message.

Captura en *eth0* y *eth1* simultánea:



Configuración persistente

Las configuraciones realizadas en los apartados anteriores son volátiles y desaparecen cuando se reinician las máquinas. Durante el arranque del sistema se pueden configurar automáticamente los interfaces según la información almacenada en el disco.

Ejercicio 12 [Router]. Crear los ficheros ifcfg-eth0 e ifcfg-eth1 en el directorio /etc/sysconfig/network-scripts/ con la configuración de cada interfaz. Usar las siguientes opciones (descritas en /usr/share/doc/initscripts-*/sysconfig.txt):

```
TYPE=Ethernet
BOOTPROTO=none
IPV6ADDR=<dirección IP en formato CIDR>
IPV6_DEFAULTGW=<dirección IP del encaminador por defecto (en este caso, no tiene)>
DEVICE=<nombre del interfaz>
```

Copiar el contenido de los ficheros.

En VM3:

```
nano /etc/sysconfig/network-scripts/ifcfg-eth0
```

```
TYPE=Ethernet
```

```
BOOTPROTO=none
```

```
IPV6ADDR="fd00:0:0:a::3/64"
```

```
DEVICE=eth0
```

```
IPV6FORWARDING=yes
```

```
nano /etc/sysconfig/network-scripts/ifcfg-eth1
```

```
TYPE=Ethernet
```

```
BOOTPROTO=none
```

```
IPV6ADDR="fd00:0:0:b::1/64"
```

```
DEVICE=eth1
```

```
IPV6FORWARDING=yes
```

Ejercicio 13 [Router]. Comprobar la configuración persistente con las órdenes `ifup` e `ifdown`.

Copiar los comandos utilizados y su salida.

En **VM3**:

`ifdown eth0`

`ifdown eth1`

`ifup eth0`

`INFO : [ipv6_wait_tentative] Waiting for interface eth0 IPv6 address(es) to leave the 'tentative' state`

`INFO : [ipv6_wait_tentative] Waiting for interface eth0 IPv6 address(es) to leave the 'tentative' state`

`ifup eth1`

`INFO : [ipv6_wait_tentative] Waiting for interface eth1 IPv6 address(es) to leave the 'tentative' state`

`INFO : [ipv6_wait_tentative] Waiting for interface eth1 IPv6 address(es) to leave the 'tentative' state`

NOTA: La salida `INFO` apareció solo en el laboratorio, cuando lo probé desde casa no aparecía

Autoconfiguración. Anuncio de prefijos

El protocolo de descubrimiento de vecinos se usa también para la autoconfiguración de los interfaces de red. Cuando se activa un interfaz, se envía un mensaje de descubrimiento de encaminadores. Los encaminadores presentes responden con un anuncio que contiene, entre otros, el prefijo de la red.

Ejercicio 14 [VM1, VM2, VM4]. Eliminar las direcciones ULA de los interfaces desactivándolos con `ip link`.

En **VM1**:

`ip address delete fd00:0:0:a::1/64 dev eth0`

`ip link set eth0 down`

En **VM2**:

`ip address delete fd00:0:0:a::2/64 dev eth0`

`ip link set eth0 down`

En **VM4**:

`ip address delete fd00:0:0:b::2/64 dev eth0`

`ip link set eth0 down`

Ejercicio 15 [Router]. Configurar el servicio `zebra` para que el encaminador anuncie prefijos. Para ello, crear el archivo `/etc/quagga/zebra.conf` e incluir la información de los prefijos para las dos redes. Cada entrada será de la forma:

```
interface eth0
  no ipv6 nd suppress-ra
  ipv6 nd prefix fd00:0:0:a::/64
```

```
nano /etc/quagga/zebra.conf
interface eth0
  no ipv6 nd suppress-ra
  ipv6 nd prefix fd00:0:0:a::/64
interface eth1
  no ipv6 nd suppress-ra
  ipv6 nd prefix fd00:0:0:b::/64
```

```
service zebra start
```

Finalmente, arrancar el servicio con el comando `service zebra start`.

Ejercicio 16 [VM4]. Comprobar la autoconfiguración del interfaz de red en VM4, volviendo a activar el interfaz y consultando la dirección asignada.

Copiar la dirección asignada.

*En **vm4**:*

`ip link set eth0 up`

`ip address`

`1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000`

`link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00`

`inet 127.0.0.1/8 scope host lo`

`valid_lft forever preferred_lft forever`

`inet6 ::1/128 scope host`

`valid_lft forever preferred_lft forever`

`2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000`

`link/ether 08:00:27:6b:8f:45 brd ff:ff:ff:ff:ff:ff`

`inet6 fd00::b:a00:27ff:fe6b:8f45/64 scope global tentative mngtmpaddr dynamic`

`valid_lft 2592000sec preferred_lft 604800sec`

`inet6 fe80::a00:27ff:fe6b:8f45/64 scope link`

`valid_lft forever preferred_lft forever`

Ejercicio 17 [VM1, VM2]. Estudiar los mensajes del protocolo de descubrimiento de vecinos:

- Activar el interfaz en VM2, comprobar que está configurado correctamente e iniciar una captura de paquetes con Wireshark.
- Activar el interfaz en VM1 y estudiar los mensajes ICMP de tipo Router Solicitation y Router Advertisement.
- Comprobar las direcciones destino y origen de los datagramas, así como las direcciones destino y origen de la trama Ethernet. Especialmente la relación entre las direcciones IP y MAC. Estudiar la salida del comando `ip maddr`.

Copiar una captura de pantalla de Wireshark.

*En **VM2**:*

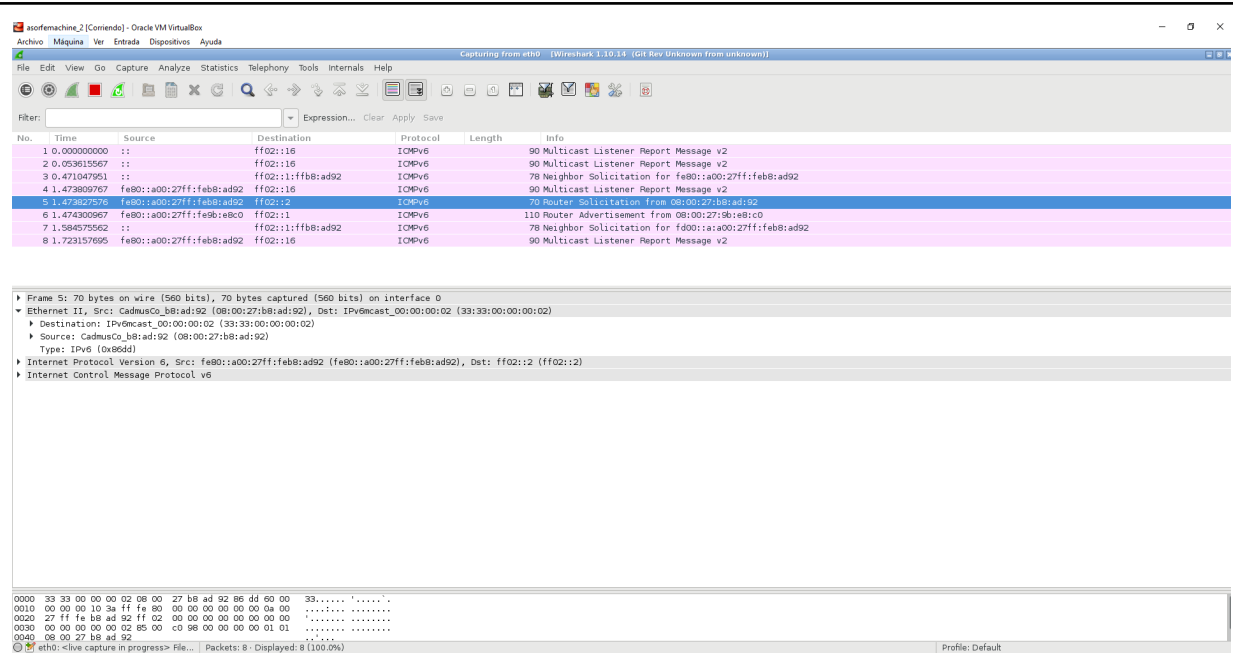
`ip link set eth0 up`

`ip a : fd00::a:a00:27ff:feea:9bf2/64`

Captura Wireshark en VM2:

*En **VM1**:*

`ip link set eth0 up fd00::a:a00:27ff:feb8:ad92/64`



Primero se asigna una dir de enlace (fe80::a00:27ff:feb8:ad92/64) a vm1

Después de **vm1** se envían **ROUTER SOLICITATION** a la dir multicast 33:33:00:00:00:02 y ff02::2

Después de **vm3** se envían **ROUTER ADVERTISEMENT** a la dir multicast 33:33:00:00:00:01 y ff02::1

En VM1: `ip maddr`

```
1: lo
   inet 224.0.0.1
   inet6 ff02::1
   inet6 ff01::1
2: eth0
   link 01:00:5e:00:00:01
   link 33:33:00:00:00:01
   link 33:33:ff:b8:ad:92
   inet 224.0.0.1
   inet6 ff02::1:ffb8:ad92 users 2
   inet6 ff02::1
   inet6 ff01::1
```

Para saber más... En el proceso de autoconfiguración se genera también el identificador de interfaz según el *Extended Unique Identifier* (EUI-64) modificado. La configuración del protocolo de anuncio de encaminadores tiene múltiples opciones que se pueden consultar en la documentación de zebra (ej. intervalo entre anuncios no solicitados). Cuando sólo se necesita un servicio que implemente el anuncio de prefijos, y no algoritmos de encaminamiento para el router, se puede usar el proyecto de código libre *Router Advertisement Daemon*, *radvd*.

Ejercicio 18 [VM1]. La generación del identificador de interfaz mediante EUI-64 supone un problema de privacidad para las máquinas clientes, que pueden ser rastreadas por su dirección MAC. En estos casos, es conveniente activar las extensiones de privacidad para generar un identificador de interfaz pseudoaleatorio temporal para las direcciones globales. Activar las extensiones de privacidad en VM1 con `sysctl -w net.ipv6.conf.eth0.use_tempaddr=2`.

Copiar la dirección asignada.

En VM1: Necesitamos borrar la que se ha asignado antes

`ip address delete fd00::a:a00:27ff:feb8:ad92/64 dev eth0`

`ip link set eth0 down`

```

sysctl -w net.ipv6.conf.eth0.use_tempaddr=2
ip link set eth0 up
ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group
default qlen 1000
    link/ether 08:00:27:b8:ad:92 brd ff:ff:ff:ff:ff:ff
    inet6 fd00::a:aa:e37c:d618:58fe/64 scope global temporary tentative dynamic
        valid_lft 604800sec preferred_lft 85800sec
    inet6 fd00::a:a00:27ff:feb8:ad92/64 scope global tentative mngtmpaddr dynamic
        valid_lft 259200sec preferred_lft 604800sec
    inet6 fe80::a00:27ff:feb8:ad92/64 scope link
        valid_lft forever preferred_lft forever

```

ICMPv6

El protocolo ICMPv6 permite el intercambio de mensajes para el control de la red, tanto para la detección de errores como para la consulta de la configuración de ésta. Durante el desarrollo de la práctica hemos visto los más importantes.

Ejercicio 19. Generar mensajes de los siguientes tipos en la red y estudiarlos con ayuda de Wireshark:

- Solicitud y respuesta de eco.
- Solicitud y anuncio de encaminador.
- Solicitud y anuncio de vecino.
- Destino inalcanzable - Sin ruta al destino (Code: 0).
- Destino inalcanzable - Dirección inalcanzable (Code: 3)
- Destino inalcanzable - Puerto inalcanzable (Code: 4)

Copiar capturas de pantalla de Wireshark con los tres últimos mensajes.

- Solicitud y respuesta de eco + Solicitud y anuncio de vecino

*En VM1: **ping6 -c 1 fd00::a:a00:27ff:feea:9bf2 -I eth0 (scope global mngtmpaddr dynamic de vm2)***

PING fd00::a:a00:27ff:feea:9bf2(fd00::a:a00:27ff:feea:9bf2) from fd00::a:aa:e37c:d618:58fe eth0: 56 data bytes

64 bytes from fd00::a:a00:27ff:feea:9bf2: icmp_seq=1 ttl=64 time=0.518 ms

--- fd00::a:a00:27ff:feea:9bf2 ping statistics ---

1 packets transmitted, 1 received, 0% packet loss, time 0ms

rtt min/avg/max/mdev = 0.518/0.518/0.518/0.000 ms

- Solicitud y anuncio de encaminador.

En VM1: ip link set eth0 down y luego ip link set eth0 up (como el ej17 wireshark en vm2)

- Destino inalcanzable - Sin ruta al destino (Code: 0)

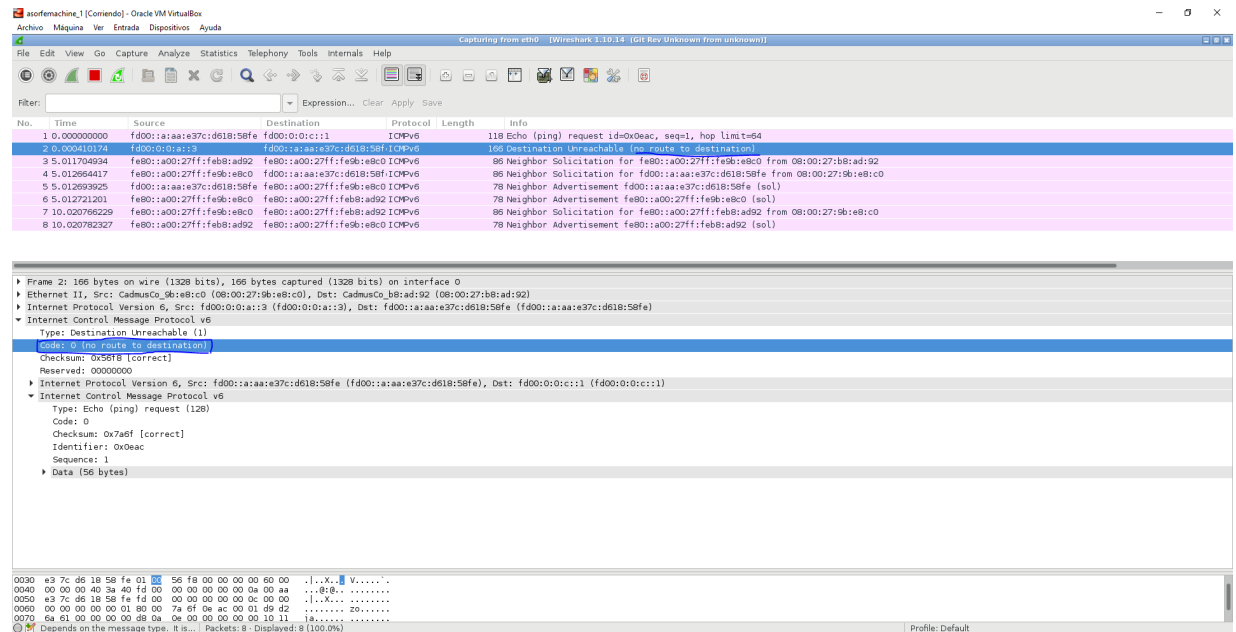
*En VM1: **ping6 -c 1 fd00:0:0:c::1 -I eth0***

PING fd00:0:0:c::1(fd00:0:0:c::1) from fd00::a:aa:e37c:d618:58fe eth0: 56 data bytes

From fd00:0:0:a::3 icmp_seq=1 Destination unreachable: No route

--- fd00:0:0:c::1 ping statistics ---

1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms



- Destino inalcanzable - Dirección inalcanzable (Code: 3)

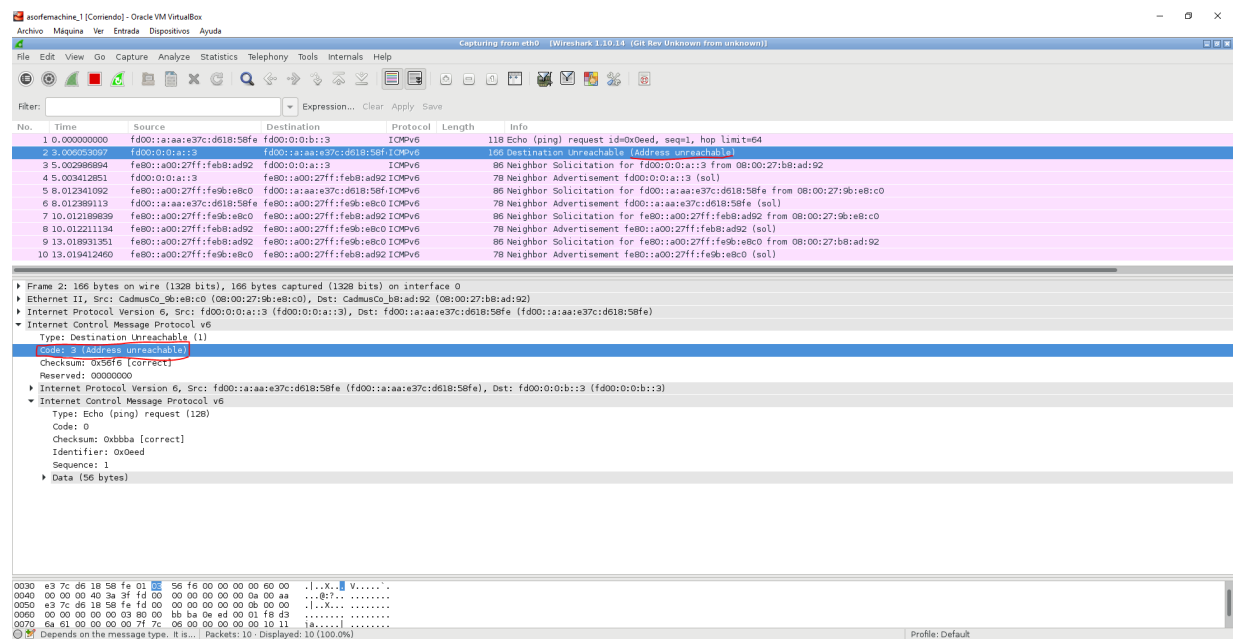
En VM1: **ping6 -c 1 fd00:0:0:b::3 -I eth0** (la antigua red de vm3 y vm4 serviría cualquier b::nº)

PING fd00:0:0:b::3 (fd00:0:0:b::3) from fd00::a:aa:e37c:d618:58fe eth0: 56 data bytes

From fd00:0:0:a::3 icmp_seq=1 Destination unreachable: Address unreachable

--- fd00:0:0:b::3 ping statistics ---

1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms



- Destino inalcanzable - Puerto inalcanzable (Code: 4)

En **VM1**: `nc -6 -u fd00::a:bd41:5687:fe03:ab5b 5` (scope global temporary dynamic de **VM2**)
 ^X(ctrl +X)

Ncat: Connection refused.

The image shows a Wireshark packet capture window. The top pane displays a list of captured packets. Packet 2, at time 0.000998669, is an ICMPv6 message of type 112 (Destination Unreachable) from source fd00::a:bd41:5687:fe03:ab5b to destination fd00::a:bd41:5687:fe03:ab5b. The bottom pane shows the detailed view of this packet. It is an Internet Control Message Protocol version 6 (ICMPv6) message. The type is 'Destination Unreachable' (112). The code is 'Port unreachable' (4). The packet contains a User Datagram Protocol (UDP) header and data. The UDP header shows source port 58854 and destination port rje (5). The data field contains 2 bytes of data.